A Hybrid Load Flow and Event Driven Simulation Approach for Multi-State System Reliability Evaluation

Hindolo George-Williams, Edoardo Patelli*

Institute for Risk and Uncertainty, Chadwick Building, University of Liverpool, Peach Street, Liverpool L69 7ZF, United Kingdom

Abstract

Structural complexity of systems, coupled with their multi-state characteristics, renders their reliability and availability evaluation difficult. Notwithstanding the emergence of various techniques dedicated to complex multi-state system analysis, simulation remains the only approach applicable to realistic systems. However, most simulation algorithms are either system specific or limited to simple systems since they require enumerating all possible system states, defining the cut-sets associated with each state and monitoring their occurrence. In addition to being extremely tedious for large complex systems, state enumeration and cut-set definition require a detailed understanding of the system's failure mechanism. In this paper, a simple and generally applicable simulation approach enhanced for multi-state systems of any topology is presented. Here, each component is defined as a Semi-Markov stochastic process and via discrete-event simulation, the operation of the system is mimicked. The principles of flow conservation are invoked to determine flow across the system for every performance level change of its components using the interior-point algorithm. This eliminates the need for cut-set definition and overcomes the limitations of existing techniques. The methodology can also be exploited to account for effects of transmission efficiency and loading restrictions of components on system reliability and performance. The principles and algorithms developed are applied to two numerical examples to demonstrate their applicability.

Keywords:
System Reliability, Event-Driven Simulation, Multi-State System, Complex System, Availability, Load-Flow

1. Introduction

A system is deemed structurally complex if it's a non-series, non-parallel or non-series-parallel interconnection of components and subsystems. Components are a system's smallest building block and therefore determine its output levels, states and behaviour. In realistic systems, these components may exist in one of several possible states/output levels [1] dictated by their failure characteristics, operating conditions, age or some stochastic event outside the system boundary. The result is a system characterised by multiple states, with the number of states moderated by diversity in output level of components and system structure [2, 3]. Unlike binary-state systems which can either be perfectly working or completely failed, multi-state systems can exist in intermediate states. The number of intermediate states may or may not be finite depending on the performance measure under consideration and the type of system being studied [3]. For instance, the power generated by a hydroelectric power plant may take any value between zero and its maximum achievable level depending on the height of water in the dam, the performance level of its components and demand on the grid. Other examples of multi-state systems are communication systems; where data processing speed [4, 5] may be the performance measure, cooling systems; where coolant flow rate or cooling capacity [6] may be the performance measure and production systems; where production rate is the performance measure. These systems may either be standalone or form an indispensable integral part of some critical system like safety critical and industrial control systems. Therefore, it is important to be able to predict their susceptibility to failure and quantify the ensuing consequences on their performance for effective planning of preventive and corrective measures. This

*Corresponding author
Email addresses: H.George-Williams@liv.ac.uk (Hindolo George-Williams), epatelli@liverpool.ac.uk (Edoardo Patelli)

process is referred to as reliability prediction. Reliability prediction has cruised through tremendous developments; moving from traditional methods that consider the failure of a system as being a consequence of failure of its components only to methods that treat failure of a system from a wider perspective including external factors [7]. Today, reliability prediction transcends just defining a set of standards for predicting the failure rate of components to system-level and safety of complex engineering systems [8].

1.1. Current System Reliability Evaluation Techniques

In system reliability and performance evaluation, the analyst has numerous techniques at their disposal. Sometimes one technique cannot quite yield the required outcome and a collection of techniques is required instead. The technique employed is determined by the system being analysed, reliability indices of interest, available computing resources and the degree of precision demanded. These techniques according to [9, 10] (cited in [2]), can be classed as either *heuristic*, *analytical* or *simulation based*. They can also be classified on the basis of applicability in which case they can either be *static* or *dynamic* [2]. Unlike static techniques, dynamic techniques do not only model the system based on functional and structural relationships between its components but also support dynamic relationships like inter component dependencies.

Reliability Block Diagrams[11, 12] and Fault Trees [11, 13] are the most common traditional reliability analysis techniques and have been used extensively in reliability evaluation of binary-state systems. Reliability Block Diagrams are a graphical expression of functional relationships between system components in terms of combination of functioning components required for system success. Fault Trees express this functional relationship via boolean logic gates and depict the combination of component failure that culminates into system failure. These two techniques prove particularly useful for moderately sized systems with series-parallel configurations but become cumbersome with large complex systems and often require additional techniques [14] to decompose the system. Overcoming this difficulty necessitated the development of Reliability Graphs. Reliability Graphs [15, 16] represent the system as a network of nodes connected by edges and they are very much efficient in modelling structural complexities; system failure is defined as the non-existence of path between source and sink nodes [2]. These three techniques utilise the assumption of components being statistically independent rendering them incapacitated for systems with dynamic properties like restrictive repairs and systems with components exhibiting dependent and dynamic relationships. However, techniques including but not limited to Dynamic Reliability Block Diagrams [2], Dynamic Fault Trees [17], Condition-based Fault Trees [18], Dynamic Flow Graphs [19, 20], Petri Nets [21] and other combinatorial techniques [22] have been developed to model these dynamic relationships. They have found application in a wide range of reliability engineering problems including repairable systems with restrictive maintenance policies [21, 22].

Though the earliest forms of these techniques were applicable only to binary-state systems, numerous instances of their recent extension to multi-state systems are present in literature. Lisnianski [23] employed an extended block diagram method to apply classical block diagram principles to a repairable system. Binary Decision Diagrams (BDD) [24, 25] which underlying principles are built upon Boolean algebra have also been applied with success to multi-state system reliability evaluation. They proceed via a state enumeration procedure in which each system state is represented by a multi-state fault tree. Unfortunately, state enumeration is only feasible for moderately sized and simple systems, for complex systems it's expensive and error prone if done by hand. Consequently, their applicability is limited to moderately sized simple systems. In recent years, reliability graphs have also attracted significant attention which interest has given birth to algorithms optimized for multi-state system reliability evaluation. Though, Yeh on two separate occasions([26] and [27]) developed algorithms that do not require prior knowledge of all minimal paths or cuts of the system, most graph based algorithms do [28–31]. As such, exploiting them requires first deriving the desired path or cut sets using other well known algorithms which in itself is an NP-hard problem [26, 27]. Compounding their challenges (including Yeh's algorithms in [26] and [27]) is the fact that they are based on the assumption that capacities of system components take only non-negative integer values. However, in many systems, components and system capacities are positive but not necessarily integer valued. In addition to their individual short falls, the extended block diagram technique, BDD and graph based algorithms share two common limitations. They define system reliability with respect to maximum flow through the system. Therefore they are limited to systems with single output nodes or systems (like signal transmission networks [32]) with multiple output nodes in which only the presence of flow at these nodes is desired but the relative magnitude is irrelevant. They stop short at solving multi-output systems with com-

peting demand at the output nodes. The second limitation arises from the assumption that there are no flow losses in the system, implying inapplicability to certain practical engineering systems encountered in real life in which flow under some condition (e.g. component failure) escapes across the component/system boundary.

Various researchers [1, 4–6, 33, 34] have made invaluable contributions to multi-state system reliability analysis, developing techniques applicable to a wide range of systems. These techniques are based mainly on either the structure function approach, stochastic process, simulation or the Universal Generating Function approach [3, 35, 36]. The most popular stochastic process employed in reliability analysis is Markov Chain which involves enumerating all possible states of the system and evaluating the associated state probabilities [36, 37]. This technique is only easily applicable to exponential transitions or distributions with simple cumulative distribution functions, requires complicated mathematics and becomes complex for large systems. The number of states in the model ranges from $m + 1$ for binary-state series systems to 2^m for binary-state parallel systems, where m is the number of components making up the system. For large multi-state systems, the number of states increases dramatically thereby rendering the model difficult to construct and expensive to compute. The Universal Generating Function(UGF) technique was introduced to address the large number of states problem the Markov Process technique is plagued with. It allows the algebraic derivation of system performance from the performance distribution of its components [3, 38]. However, both techniques are limited in the number of reliability and performance indices they can quantify. In [39], Lisnianski introduced the L_z-Transform and inverse L_z-Transform concepts to enhance the derivation of instantaneous reliability measures like system reliability *R(t)*, instantaneous availability *A(t)* and instantaneous system output *X(t)* with the UGF technique [6, 38]. The UGF is a powerful tool, its applicability has been extended even to systems with dependent components as illustrated by Levitin [35, 40]. However, like the other multi-state system reliability evaluation techniques previously discussed, the UGF is maximum flow based and assumes flow conservation across system components. Also, though straight forward for systems with simple series/parallel architecture, it requires substantial effort for systems with a non-series/parallel structure including those composed of many subsystems. These considerations have hindered its application to certain multi-state system reliability problems.

Simulation techniques [41–43] are the most suitable for multi-state system reliability and performance evaluation since they mimic the actual operation of the system. Their advantages over other techniques are derived from the fact that they can support any transition distribution type, enhance the study of effects of external uncertainties on system performance and reliability [43] and are easily integrated with other reliability analysis techniques [44, 45]. Though computationally expensive for large systems and small failure probabilities, techniques that reduce the computation time and effort now exist, thanks to recent advances in computing. Variance reduction techniques and parallel computing can reduce the computation time and effort by substantial amounts when adopted. Also in extensive use are Subset Simulation [46] and Line Sampling [47, 48], both of which improve the efficiency of simulations. Instances of simulation algorithms relying on prior knowledge of path or cut-sets are found in [43, 49, 50]. Yeh et al. [51] however developed a simulation approach that doesn't require prior knowledge of system path/cut sets but applicable only to binary-state systems.

Analytical techniques exhibit superior computational efficiency over their simulation counterparts. However, as already expressed, they suffer a setback when solicited for realistic systems. Often, the reliability analyst is not only interested in steady-state or instantaneous reliability measures but also the underlying probability distributions governing the behaviour of the system as well as effects of external uncertainties (e.g., restrictive maintenance schemes, human, environmental and other stochastic external factors) on the values assumed by these measures. When faced with such scenarios, simulation becomes the most feasible alternative. In summary, both analytical and simulation procedures are restricted by their various unique limitations. Therefore, the development of a single technique that addresses these is vital.

1.2. Proposed Approach

The proposed approach is based on the fact that if the properties of components building a system are known, then its output performance can be deduced directly from its network model. Implementing this necessitates modelling each component as a multi-state object characterised by a state space diagram depicting its possible states with their interactions and a set defining the properties associated with each state. The operation of the system is simulated by generating component states and transition times from their state-space representation. The generated transitions are effected and the system output performance evaluated at each transition. Transition and output histories of components as well as the

system are captured and saved as the simulation progresses. These enhance the evaluation of any reliability/performance index both at the system and component levels. To replicate the actual operating procedure of systems, a special component *shut down* and *restart* operation is instituted. In this operation, the availability of each system component is tested at every transition against its predefined reference minimum input/output load level. With this, the latent interdependence of system components is accounted for.

A common feature exhibited by components of many realistic systems is *transmission inefficiency*; a term ascribed to the phenomenon in which an intermediate component transmits only a fraction of flow received from its preceding component. The component in other words acts as a partial sink and dissipates part of the flow such that the flow transmitted to the next component is less than that received. Under this condition, flow is no longer conserved and techniques built around the flow conservation principle become obsolete. To illustrate the effect of this phenomenon on system reliability, consider a 50MW power generator supplying a 45MW load through a 75MW transformer. If there are no power losses in the transformer, 45MW will be transmitted to the load. However, if the capacity of the transformer remains unchanged but its efficiency reduces to 75%, it takes 50MW from the generator (assuming capacity constraint is imposed) but transmits only 37.5MW to the load. In both cases, the apparent difference between capacity and demand remains constant but the power drawn from the generator increases and the effective power supplied to the load reduces. Other examples are a power transmission line prone to losses and an oil pipeline in which a mode of failure could be a hole in a pipe or gasket failure at some flange.

The transmission inefficiency of system components may exert undesirable effects on system performance and reliability, it's therefore worthwhile considering this property in the reliability evaluation process. In the scenario just discussed, the generator would need to be rated at least 60MW to match demand. Therefore, if the system is not equipped with adequate controls such that the capacity constraint is always satisfied, the generator may fail due to overloading even though demand remains well below its rated capacity. Considering this in the proposed methodology, each component state is defined by an associated output *performance level* and *sink index* which respectively state the maximum flow accepted or generated and the proportion dissipated when the component resides in that state.

Convenient representation of system architecture and evaluating system output from changes in states of components are the two prime difficulties encountered in simulation of complex multi-state systems. In the proposed approach, these are overcome by applying the concept of adjacency matrices from network theory to define the structure of the system. Adjacency matrices are a square array of 1's and 0's depicting the connectivity of a network and can easily represent any system architecture. They can be manipulated to obtain system flow equations which are solved to determine the magnitude of flow through every node of the system. The efficiency of this lies in the fact that it eliminates the need for definition of cut sets or enumeration of system states; both of which normally become laborious with complex systems. Complex network theory is a widely used concept and finds application in many engineering and real life problems. It's particularly useful in representing and analysing complexities in system structure. As a result many researchers have applied its principles to a variety of problems, yielding excellent results. For instance, Dwivedi et al.[52] used it in vulnerability analysis of a power system, Todinov[53] established his optimization of repairable flow networks entirely on it and Chen et al. [54] used it to perform a reliability/availability analysis on Manhattan street networks.

1.2.1. Advantages Over Existing Techniques

The following enumerates the key contribution of the proposed technique to multi-state system reliability evaluation.

1. Being simulation based, it inherits all the advantages of simulation approaches to system reliability and performance evaluation. With respect to other simulation algorithms, it can implement any system structure with relative ease since it doesn't require knowledge of minimal path or cut sets prior to system analysis.
2. It is not maximum flow based. Instead, it calculates the actual flow across every node of the system. This consequently makes possible the following;
   a). analysis of systems with multiple source and sink nodes with competing demand which can be static or instantaneous
   b). analysis of systems prone to losses at nodes and across edges
   c). easy restart and shut-down of components so as to replicate the actual operation of systems where necessary.
3. Node capacities and system demand can take on any positive value. They do not necessarily have to be integer valued as required by graph based algorithms.

1.3. Outline of Paper

The remainder of this paper is organised into four sections. Section 2 describes the modelling of system components followed in Section 3 by a description of how the system is modelled. The latter also contains details of the simulation procedure and its associated limitations. Implementation of the numerical case studies to show the applicability of the approach and an analysis of its computational requirements are contained in Section 4. Section 5 constitutes the conclusion.

2. Component Modelling

Multi-state components and systems are discrete in space but continuous in time. That is, they exist in only one state at any given instance but reside continuously in that state until a transition occurs. The transition instantaneously takes them to another state where they also reside continuously until the next transition [11]. In components, these transitions are defined by time dependent probability distributions or some stochastic event outside the component boundary which underlying probability distribution may or may not be known.

Interactions between the states of an n state component and their corresponding transition probability density functions can be represented by an n order square *Transition Matrix*, **T**. Sometimes certain transitions may be controlled by a direct effect of events outside the component boundary. A renown example of this would be a system composed of a series connection of multiple repairable components.

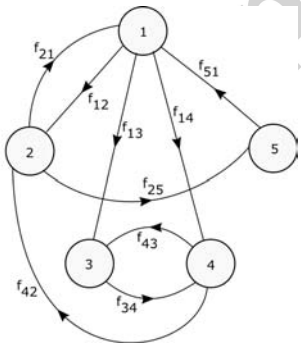


Figure 1: State space diagram of an arbitrary multi-state component

On failure of a component, the other operating components would have to be shut down and restarted only after the failed component is restored. In such a system, the shut down and restart of operating components is triggered by the failure and repair of another component. However, the occurrence of these failure and repair events is marked by uncertainty, making it impossible to assign a probability distribution to component transitions triggered by them. Also, the triggered transitions may not exhibit the Markovian property, in that the component's next state can also depend on its previous state(s). These transitions will be referred to as *forced transitions* since they are induced by events outside the component boundary and *normal transition* otherwise.

If the transition from state x to y is represented by the pair (x, y) and determined by its probability density function $f_{xy}(t)$, then each element at position (x, y) of **T** is equal to $f_{xy}(t)$. Assuming all possible transitions from state x have the same priority, the next state, y depends only on which transition occurs first. Hence, **T** completely defines the stochastic behaviour of the component as outlined in Equation 1.

$$\mathbf{T} = \{f_{xy}(t)\}_{n\times n} \mid x \neq y \quad (x, y) \in \{1, 2, ..., n\}$$
$$\mathbf{T}(x, y) = \begin{cases} \infty, & \text{If (x,y) is a forced transition} \\ 0, & \text{If no transition between x \& y} \\ f_{xy}(t), & \text{Otherwise} \end{cases} \tag{1}$$

Figure 1 shows the state space diagram of an arbitrary five-state component, where the label beside each arc represents the probability density function of the transition depicted.

The performance level; c_x of a binary-state component in state x is such that $c_x \in \{0, c_{max}\}$ where $c_x = c_{max}$ if the component is working and *0* otherwise. For multi-state components, the performance level is defined by the set $0 \leq c_x \leq c_{max}$, where c_{max} is its maximum performance level. Each state is characterised by a maximum value of load; c_x the component can supply or transmit and is referred to as its capacity in that state. Therefore, the performance of a component is defined by the vector, **C** of state capacities as follows,

$$\mathbf{C} = \{c_x\}^n \mid 0 \leq c_x \leq c_{max}, \quad n \geq 2 \tag{2}$$

The *sink index*; ε_x is introduced to define the amount of flow dissipated in the component in state x as a fraction of the total flow received. Applicable only to intermediate components, this property of the component takes a value between 0 and 1. A value of 0 meaning outflow is equal to inflow and a value of 1 corresponding to the case when outflow is zero but inflow greater than zero. At a value of 1, the component effectively becomes a sink, which explains the choice of name for this property. The sink index of a component is expressed by the

vector **S**,

$$\mathbf{S} = \{\varepsilon_x\}^n \mid 0 \leq \varepsilon_x \leq 1 \tag{3}$$

Also, a component may be subjected to loading restrictions in a bid to preserve its reliability and/or ensure its safe operation. This often requires the load to exceed a threshold value but lie within the maximum load rating of the component. Outside this range, the component is either shut down or considered failed. The maximum load rating corresponds to the maximum value of **C** but the threshold load rating may be greater than its minimum value since the component may be prone to complete failures or maintenance states characterised by a c_x value of 0. Therefore, the minimum load rating of the component is defined by $\Lambda \mid 0 \leq \Lambda \leq c_{max}$.

Each parameter discussed uniquely identifies a component, determines its behaviour and subsequently its effect on system performance and reliability. Therefore, a component is represented by the quintuple; E, defined as,

$$E = (\mathbf{T}, \mathbf{C}, \mathbf{S}, \Lambda, x_0) \tag{4}$$

Where x_0 is the initial state of the component.

2.1. Application to Repairable Multi-state Components and Systems

A simple illustration will be used to describe the modelling of a multi-state component with one or more repairable partial failure modes. Consider a 40MVA generator subjected to a 20% minimum load restriction and assume it generally exists in three possible states defined as follows;

- State 1, depicting operation at its nominal output level
- State 2, depicting operation below its nominal level, say 25MVA as a consequence of its partial failure
- State 3, depicting its total failure.

Presented in Figure 2 is a representation of the interactions between the states of the generator. State 2 is the state of interest in this illustration and the objective is to explore the possible events embedded in the restoration process from *partial failure* (i.e, transition (2, 1)).
The figure is based on the assumption that the generator remains in operation whilst undergoing repairs from state 2 (on-line repairs). However, this is not the case for components of many real world engineering systems. Most components would need to be taken out of operation before repair actions can be completed. This event may also trigger the unavailability of other components of the system as explained in the preceding section. Therefore, the assumption may result in over/under estimation of certain reliability indices when wrongly applied.

Now, consider the assumption that the generator is in series with an external breaker which can undergo failure and repair and a further assumption that the former cannot be repaired on-line. To account for the period when it is out of operation as a result of failure of the breaker, a new state; 5 is introduced as shown in Figure 3. The restoration of the generator from state 2 to 1 may follow one of two possible operational principles;

- Case 1: Repair is initiated as soon as it enters state 2. This technically means the generator does not exist in states 2 and 3
- Case 2: Repair is delayed until the generator is not in operation i.e., either totally failed or when shut down from partial failure.

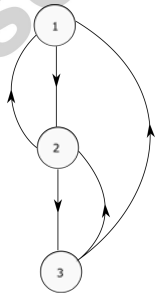
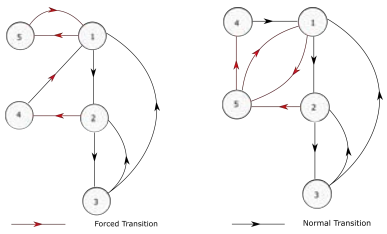


Figure 2: State space diagram of 40MVA generator



(a) Immediate repair (b) Repair only after shut down

Figure 3: Alternative state representation of generator

These necessitate the introduction of a fifth state; 4 to account for the period when it is undergoing repairs from state 2 to 1. The new state-space representations of the generator are presented in Figure 3 showing normal and forced transitions. Forced transitions would have to

be manually effected during the simulation as they do not always only depend on the current state of the component. For instance, the generator goes to state 4 from state 5 if its previous state is state 2 else it goes to state 1 as shown in Figure 3b. Using case 2 as example, the parameters of the generator would be as given below,

$$\mathbf{T} = \begin{pmatrix} 0 & f_{12}(t) & 0 & 0 & \infty \\ 0 & 0 & f_{23}(t) & 0 & \infty \\ f_{31}(t) & f_{32}(t) & 0 & 0 & 0 \\ f_{41}(t) & 0 & 0 & 0 & 0 \\ \infty & 0 & 0 & \infty & 0 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 40 & 25 & 0 & 0 & 0 \end{pmatrix} \quad \Lambda = 8$$

The sink index for each state is zero since the generator is a source. As a rule-of-thumb, sink-index is not required for sources and sinks of a system as evident in the system flow equations derived in Section 3.2.1.

The modelling technique described can be extended to components with multiple partial failure modes. This is achieved by introducing two states (one each for shut down and repair) for each partial failure state as shown in Figure 3.

2.2. Determining Component State Transition Parameters

The key to reliability evaluation of systems composed of multi-state components is being able to correctly determine the next state of a component and when the transition takes place. The proposed algorithm for sampling the next transition parameters (next state and transition time) of a multi-state component is based on the assumption that the next state of the component depends on only its current state. Starting with the component in state x at time t_c, the algorithm is summarised as follows;

Step 1. Locate all non-zero elements in row x of $\mathbf{T}$, saving their associated column number.

Step 2. Sample each element in step 1 and save the sampled value. Deterministic elements do not need to be sampled. For instance, if $\mathbf{T}(x, y) = 10$, then 10 becomes the sampled time for transition (x, y).

Step 3. Find the minimum of the transition delay times obtained in step 2 and define a set containing the transitions with delay time equal to this minimum value.

Step 4. If the set defined in step 3 has only one element, then the next state; y is the column in $\mathbf{T}$ corresponding to the transition defined by the element. Otherwise, an element is randomly selected according to a uniform discrete distribution between 1 and the maximum number of elements in the set. The next state is deduced from the column to which this element belongs.

Step 5. The next transition time of the component is the sum of the minimum time obtained in step 3 and the current simulation time.

Algorithm 1 Sampling procedure for transition parameters of a multi-state component

Require: x and t_c are respectively the current state of the component and current simulation time

function SAMPLE(x,$\mathbf{T}$,t_c)
  $Y \leftarrow find(\mathbf{T}(x, :) > 0)$ ▷ possible transitions from state x
  $f \leftarrow \mathbf{T}(x, Y)$ ▷ Transition Distributions
  $k \leftarrow$Number of elements in f
  for $n \leftarrow$1 to k **do** ▷ Loop over number of elements
    $Delay(n) \leftarrow$Sample from $f(n)$
  end for
  $SampledTime \leftarrow min(Delay)$ ▷ get least delay
  $p \leftarrow find(Delay \leftarrow SampledTime)$ ▷ get transitions with Delay times equal to $SampledTime$
  if length of p > 1 **then** ▷ Check length of p
    $u \leftarrow rand()$ ▷ Generate a uniform random number between 0 and 1
    $index \leftarrow \lceil u * numel(p) \rceil$ ▷ Product of u and the length of p approximated to the smallest following integer
  else
    $index \leftarrow p$
  end if
  $y \leftarrow Y(index)$ ▷ get next component state
  $NextTransitionTime \leftarrow SampledTime + t_c$
  return (y, $NextTransitionTime$)
end function

These steps are summarised as a pseudo-code by Algorithm 1. Its major advantage is that it ensures an unbiased determination of the transition parameters of components including those exhibiting both deterministic and probabilistic state transitions. It's clear the algorithm will never select a *forced transition* over one exhibiting markovian properties. As a rule-of-thumb, it is not applied when the component resides in states from which only *forced transitions* are possible (e.g, state 5 in Figure 3). The simulation algorithm should therefore be equipped with special routines to force these transitions.

3. System Modelling

In complex network theory, system topology is defined by a graph, which is a set of nodes connected by edges or links across which some controlled information is transmitted. This controlled information may be the current flowing in a circuit, energy generated from a power plant, liquid pumped from a reservoir, traffic flow rate in a street network or any quantifiable quantity of interest. In a graph, there is a set, s of nodes generating the controlled information and another set, t consuming or utilising the generated information. Nodes belonging to s are *source nodes* and nodes belonging to t; *sink or target nodes*. Between these two are nodes helping in transmission of the controlled information and facilitate communication between source and sink nodes. These are called *intermediate nodes*. The success and quality of the communication process is influenced by the performance of source and intermediate nodes. If information can flow in only one direction through every edge, then the graph is said to be *directed* otherwise it's *undirected or bidirectional*.

3.1. The System as a Directed Graph

As already established in section 1, Engineering systems are designed to accomplish a specific often quantifiable process. Process flow may be possible in any direction depending on the connectivity of the system and properties of its components but at any given time, its direction is known and fixed [52]. Systems are composed of a collection of components performing different but specific roles and together determine the success of the process. Some of these components are responsible for initiating the process whilst some only serve as links to ensure progression of the process. There's also another actor normally external to the system that utilises the process and drives it through the system. The set of components initiating the process are analogous to sources in a graph, so are the components serving as links analogous to intermediate nodes and the external factor driving the process analogous to sinks. Hence, the topology of the system can be conveniently and accurately represented by a directed graph.

Since the aim in system reliability evaluation is to investigate the effects of component failure on system performance and life span, the structure of the system can be represented by a directed graph with components and output points being considered nodes connected by perfectly reliable edges i.e., edges do not fail. When modelling systems with unreliable links e.g, power distribution networks, then each unreliable link should be treated as a component and represented as a node. If G is a directed graph, then the structure of the system is defined by G as,

$$G = (\mathbf{V}, \mathbf{A}) \tag{5}$$

where $\mathbf{V}$ is the set of nodes and A, the adjacency matrix. The adjacency matrix is an M order square matrix defining the connectivity of nodes, where M is the total number of nodes of the system . A connection between node i and j in the system is represented in the adjacency matrix by 1 at the intersection of row i and column j if process flow is from node i to j ($i \rightarrow j$), 1 at the intersection of row j and column i if flow is from node j to i ($i \leftarrow j$) and 0 if a connection does not exist. The node pair, (i, j) representing a connection and flow from node i to j is known as an edge, e_{ij} of the system. A row of the adjacency matrix depicts the source node of an edge and a column; the incident/target node of the edge. The edges of the system are defined by a k by 2 matrix; e, where k the total number of edges is equal to the sum of the elements of the adjacency matrix.

$$\mathbf{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} 1 & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{6}$$

$$\mathbf{e} = \{i, j\}_{k\times 2} \mid k = \sum_{j=1}^{M}\sum_{i=1}^{M} a_{ij} \quad \forall (i, j) \in \mathbf{V} \tag{7}$$

For some systems, the links may be reliable but inefficient which can have negative effects on system reliability and performance. Let α_{ij} be the efficiency of edge e_{ij} such that $0 < \alpha_{ij} \leq 1$. Note that if the efficiency of a link is zero, then it becomes a sink and should therefore be treated as a node. If in defining the adjacency matrix, the efficiencies of the links are used, then Equation 6 becomes;

$$\mathbf{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} \alpha_{ij} & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{8}$$

and k redefined as the total number of non-zero elements of $\mathbf{A}$ i.e., $k = \sum_{j=1}^{M}\sum_{i=1}^{M}(a_{ij} > 0)$. Another property of links encountered in many real world systems (e.g, transmission lines in power distribution systems) is their maximum load rating. Let l_{ij} be the maximum allowable load for edge e_{ij} such that $0 < l_{ij} \leq \infty$. The upper bound corresponds to the case when no load restrictions are imposed on the edge. Though unrealistic, this assumption is useful in maximum flow analysis of distribution networks. To account for this third property of a network, the *capacity matrix*; $\mathbf{L}$ is introduced to define the maximum capacity of edges. Each non-zero element of $\mathbf{L}$ corresponds to a non-zero entry in $\mathbf{A}$.

$$\mathbf{L} = \{l_{ij}\}_{M\times M} \tag{9}$$

Therefore, Equation 5 can be adapted to define both the structure of a system and the properties of its links as expressed in Equation 10.

$$G = (\mathbf{V}, \mathbf{A}, \mathbf{L}) \tag{10}$$

Now that the adjacency and edge matrices have been discussed and their mathematical relations expressed, a fourth matrix defining the relationship between edges and nodes of the system with respect to the direction of process flow will be introduced. This matrix is known as the incidence matrix; $\mathbf{\Gamma}$ and is an M by k matrix with nodes represented by rows and edges by columns. If the edges are numbered from 1 to k and nodes from 1 to M, then for an edge e_{ij}, the element in the column corresponding to the edge number (i.e., index of (i,j) in matrix $\mathbf{e}$) and row i (the local source node), is assigned the value 1, the element in row j (the incident/local target node) is assigned the value $-a_{ij}$ and 0 in all other rows.

$$\mathbf{\Gamma} = \{\gamma_{pq}\}_{M\times k} \mid \gamma_{pq} = \begin{cases} 1, & p = i \\ -a_{ij}, & p = j \\ 0, & \text{otherwise} \end{cases} \qquad \forall (i,j) \in \mathbf{e} \tag{11}$$

The variable $q = 1, 2, ..., k$ (the edge number) is the index of edge (i, j) or e_{ij} in $\mathbf{e}$ and $p = 1, 2, ..., M$. Given the adjacency matrix alone of the system, $\mathbf{e}$ and $\mathbf{\Gamma}$ can be derived by applying equations 7 and 11. This procedure is summarised in Algorithm 2.

3.2. System Representation and Flow Analysis

As discussed in section 3.1, a system can be represented by a directed graph comprising source, intermediate and sink nodes. The source and intermediate nodes are physical components making up the structure of the system while sink nodes are output points via which system output is consumed. All three node types can be modelled by the methodology proposed in section 2. However, $\mathbf{T}$ is usually unknown for output nodes prior to analysis can therefore not be defined. Also, $\mathbf{S}$ is not required for output/sink nodes as evidenced in the system flow equations derived later in Section 3.2.1. Therefore, for output nodes, equation 4 could be re-written as,

$$E = (\mathbf{C}, \Lambda) \tag{12}$$

Since it's impossible to determine all output levels prior to analysis for some systems, specifying $\mathbf{C}$ for sink/output nodes is optional. Alternatively, it could be defined by specifying only the output levels of interest.

Algorithm 2 Procedure for deriving the *edge* and *incidence* matrices of a system

Require: A, the adjacency matrix of the system

function GetMatrix(A)
  $M \leftarrow size(A, 1)$ ▷ get number of nodes
  $k \leftarrow \sum_{j=1}^{M} \sum_{i=1}^{M} (a_{ij} > 0)$ ▷ get number of edges
  for $n \leftarrow$1 to M **do**
    $index \leftarrow$columns with non-zero entries
    $w \leftarrow$number of elements in $index$
    $\mathbf{e}(end + 1 : end + w, :) \leftarrow [n * \{1\}_{w\times 1} \quad index^T]$
  end for
  $\mathbf{\Gamma} \leftarrow \{0\}_{M\times k}$ ▷ predefine the incidence matrix by an M by k array of zeros
  $i \leftarrow$vector of elements in column 1 of $\mathbf{e}$
  $j \leftarrow$vector of elements in column 2 of $\mathbf{e}$
  $position \leftarrow j^T + \{0, 1, ..., k-1\} * M$
  $position1 \leftarrow i + (j-1) * M$
  $\mathbf{\Gamma}(position) \leftarrow -\mathbf{A}(position1)$ ▷ update values
  $position2 \leftarrow i^T + \{0, 1, ..., k-1\} * M$
  $\mathbf{\Gamma}(position2) \leftarrow 1$ ▷ update values
  return ($\mathbf{e}$,$\mathbf{\Gamma}$)
end function

If $\mathbb{E}$ is the set containing the properties, E_i of each node of the system, then the system structure and property can be defined by the set $\mathbb{S}$.

$$\mathbb{S} = (G, \mathbb{E}) \mid \mathbb{E} = \{E_i\}^M \quad \forall i \in \{1, 2, ..., M\} \tag{13}$$

3.2.1. Derivation of System Flow Equations

To understand the flow of processes in systems, it's worthwhile to think of source nodes as reservoirs and intermediate nodes as valves in a pipe network. The total flow through the system depends on the amount of liquid in the reservoir and the resistance to flow in the pipe network. These properties are functions of the capacities of the source and intermediate nodes respectively. As source and intermediate nodes perform their functions, their capacities change consistent with their state-space diagrams, leading to a change in resistance of some flow paths. The result is the existence of paths with high resistance and some with very low resistance triggering differential flow redirection. The amount of flow through a path is directly proportional to the capacity of the smallest node in that path. Hence, more flow will be redirected through the path of least resistance (highest capacity) as illustrated in figure 4.

Node 1 is a source, node 5 a sink and nodes 2, 3 & 4 are intermediate nodes. Initially, all intermediate nodes have the same capacity, resulting in equal path resis-

tance for both upper and lower paths between source and sink. Hence, the 1 unit of flow from the source is split equally between the paths as shown in Figure 4a. In Figure 4b, the capacity of node 4 reduces to 0.2 thereby creating a difference in resistances of the two paths. Since the maximum capacity of the upper path is 1, 0.8 units of flow are redirected through it and 0.2 through the lower path. If at this stage, the capacity of node 3 is also reduced to 0.5 (Figure 4c), the capacity of the upper path becomes 0.5. Therefore, only 0.5 units of flow are transmitted along the upper path and 0.2 along the lower.

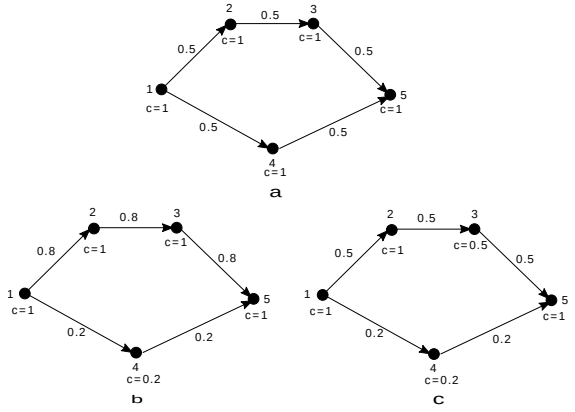


Figure 4: Flow visualisation in an arbitrary 5 node system

Flow redirection may appear simple and straight forward for systems as simple as the one shown in Figure 4 but presents itself as an optimization problem for large systems with complex topology. The optimization normally consists of multiple parameters and constraints and has as its goal the calculation of flow along each edge of the network.

Let X_{ij} be the magnitude of flow in edge e_{ij}, $\mho_i^+$; the set of nodes connected to the inlet of node i, $\mho_i^-$; the set of nodes connected to the outlet of node i, $c_x^{\{i\}}$; the current capacity of node i and x; the current state of node i. The total inflow for source nodes is zero and for sink nodes, the total outflow is zero which means $i \in s$ if $\mho_i^+ = \varnothing$ and $i \in t$ if $\mho_i^- = \varnothing$. The constraints and by extension the equations governing the optimization procedure hinge on the following assumptions;

1. The system is equipped with adequate controls such that flow does not exceed the capacity of a node. This is known as the capacity constraint. For intermediate and sink nodes, it means the total inflow should not exceed the node capacity and it's expressed mathematically as,

$$\sum_{j \in \mho_i^+} X_{ji}\alpha_{ji} \leq c_x^{\{i\}} \mid (i,j) \in \mathbf{e}, \quad \mho_i^+ \subset \mathbf{V} \tag{14}$$

For source nodes, the statement means the total outflow should not exceed the node capacity;

$$\sum_{j \in \mho_i^-} X_{ij} \leq c_x^{\{i\}} \mid (i,j) \in \mathbf{e}, \quad \mho_i^- \subset \mathbf{V} \tag{15}$$

Equations 14 and 15 can be combined into a single equation defining the capacity constraint of all the nodes of the system.

$$\mathbf{\Theta}\{X_{ij}\}_{k\times 1} \leq \{c_x^{\{i\}}\}_{M\times 1} \mid (i,j) \in \mathbf{e}, \quad \forall i \in \mathbf{V} \tag{16}$$

The matrix, $\mathbf{\Theta}$ is related to the incidence matrix, $\mathbf{\Gamma}$ of the system as follows,

$$\mathbf{\Theta} = \{\theta_{iq}\}_{M\times k} \mid \theta_{iq} = \begin{cases} \gamma_{iq}, & i \in s \\ -\gamma_{iq}, & \gamma_{iq} < 0 \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

2. Flow in the system is conserved. That is,
   - Total flow generated by sources equals the sum of flow consumed by sinks and any losses at intermediate nodes.
   - The total inflow at an intermediate node; $i \mid i \in (s \cup t)'$ equals its total outflow plus any losses at the node.

Expressing the second statement mathematically, equation 18 is obtained;

$$\sum_{j \in \mho_i^-} X_{ij} - \sum_{j \in \mho_i^+} X_{ji}\alpha_{ji} = 0 \mid (i,j) \in \mathbf{e} \tag{18}$$

The flow conservation constraint equation for the system can be obtained by applying equation 18 to all intermediate nodes of the system. This equation is defined as,

$$\mathbf{\Phi}\{X_{ij}\}_{k\times 1} = \{0\}_{\eth\times 1} \quad \forall (i,j) \in \mathbf{e} \tag{19}$$

where $\eth$ is the number of intermediate nodes. If the incidence matrix, $\mathbf{\Gamma}$ is expressed in terms of its rows, Γ_p i.e.,

$$\mathbf{\Gamma} = \begin{pmatrix} \Gamma_1 \\ \Gamma_2 \\ . \\ . \\ . \\ \Gamma_{M-1} \\ \Gamma_M \end{pmatrix} = \{\Gamma_p\}^M \mid p = 1, 2, ..., M \tag{20}$$

then $\mathbf{\Phi}$ and $\mathbf{\Gamma}$ are related as expressed in equation 21.

$$\mathbf{\Phi} = \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ . \\ . \\ . \\ \Phi_{\eth-1} \\ \Phi_{\eth} \end{pmatrix} = \{\Phi_\lambda\}^{\eth} \mid \Phi_\lambda = \Gamma_p, \quad \lambda = 1, 2, ..., \eth$$
$$\eth < M, \quad f : \lambda \rightarrow p \quad \forall p \in (s \cup t)' \tag{21}$$

In other words, $\mathbf{\Phi}$ (row designated as Φ_λ) is a $\eth$ by k sub matrix of $\mathbf{\Gamma}$ containing all the rows in $\mathbf{\Gamma}$ corresponding to intermediate nodes.
If the transmission efficiency of intermediate nodes is considered, and given that $\varepsilon_x^{\{i\}}$ is the current sink index of node i, equation 18 is rewritten as,

$$\sum_{j \in \mho_i^-} X_{ij} - (1 - \varepsilon_x^{\{i\}}) \sum_{j \in \mho_i^-} X_{ji}\alpha_{ji} = 0 \mid (i, j) \in \mathbf{e} \tag{22}$$

and the matrix, $\mathbf{\Phi}$ in equation 19 redefined as

$$\mathbf{\Phi} = \{\phi_{\lambda q}\}_{\eth \times k} \mid \phi_{\lambda q} = \begin{cases} (1 - \varepsilon_x^{\{p\}})\gamma_{pq}, & \gamma_{pq} < 0 \\ \gamma_{pq}, & \text{otherwise} \end{cases}$$
$$\lambda = 1, 2, ..., \eth, \quad \eth < M$$
$$f : \lambda \rightarrow p \quad \forall p \in (s \cup t)' \tag{23}$$

Comparing equations 21 and 23 reveals that the former is a special case of the latter which is the case when the value of $\varepsilon_x^{\{i\}}$ is set to 0 (i.e., 100% efficient nodes).

3. If the capacity of a node changes, the flow in the system is reconfigured to match the prevailing system conditions. The flow,$\mathbf{\Psi}$, generated by sources is such that flow into sinks is maximised. That is, sources always try to match available demand. For a system, $\mathbf{\Psi}$ is defined as,

$$\begin{aligned} \mathbf{\Psi} &= -\sum_{j \in \mho_i^-} \sum_{i \in s} X_{ij} \\ &= -\{\psi_q\}_{1 \times k} \{X_{ij}\}_{k \times 1} \mid \psi_q = \sum_{i \in s} \gamma_{iq} \\ q &= 1, 2, ..., k \end{aligned} \tag{24}$$

In the optimization problem, the goal is to determine the minimum value of $\mathbf{\Psi}$ that maximises the flow in the system. Hence, $\mathbf{\Psi}$ represents the objective function of the optimization.

4. The minimum allowable flow through an edge e_{ij} is 0 but the maximum is defined by the smallest of the maximum capacities of its nodes i & j and its capacity; l_{ij}. That is, $0 \leq X_{ij} \leq \Omega_{ij}$, where Ω_{ij} is the maximum flow through the edge. If lb represents the vector of lower bounds and ub the vector of upper bounds of all edges of the network, then,

$$\begin{aligned} lb = \{0\}_{k \times 1}, \quad ub = \{\Omega_{ij}\}_{k \times 1} \\ \Omega_{ij} = \min\{c_{max}^{\{i\}}, c_{max}^{\{j\}}, l_{ij}\} \quad \forall (i, j) \in \mathbf{e} \end{aligned} \tag{25}$$

Equations 16, 19, 24 and 25 form the basis of the optimization process which can be implemented by a variety of well known algorithms. However, the numerical examples presented in this paper are based on the interior-point algorithm [55, 56] available in Matlab's linear programming toolbox.

So far, only unidirectional links have been considered. Without loss of generality, the equations proposed remain valid for bidirectional links. Such links are normally represented by two reciprocal edges i.e., edges connecting the same pair of nodes but allowing flow in opposing directions. For instance, edges e_{12} and e_{21} are reciprocals. The outcome of the optimization sometimes results in flow across both reciprocal edges. However, since in practice both edges represent the same physical link/element and given that flow at any instance is possible in only one direction, the flows should be normalised to obtain the actual flow through the link. The actual direction of flow across the link is given by the direction of the larger flow. The magnitude of this flow is obtained by temporarily setting the capacity of the edge with the smaller flow to zero and repeating the optimization process. This is expressed in Equation 26 as;

$$\Omega_{gh} = 0 \mid (g, h) = \begin{cases} (j, i) & \text{If } X_{ij} > X_{ji} \\ (i, j) & \text{Otherwise} \end{cases} \tag{26}$$

3.2.2. Output Calculation and Node Reconfiguration

In an engineering system, failure or change in output level of one component may lead to change in output level of other components. This change may trigger shut down of operating components or restart of components already in shut down. Therefore as components undergo their normal failure and repair cycles, the system also undergoes a series of *shut down* and *restart* operations. It's imperative that these shut down and restart operations be accounted for in the simulation process for a realistic outcome as the failure of most components depends on the time spent in operation.

Taking a component out of operation affects all components connected to it and hence process flow in the system. Therefore, the effective output of a node after every system transition is the flow through it when the last shut down operation has been executed and there are no nodes left requiring shut down. For this reason, an iterative procedure is employed in system output calculation. This procedure is based on the following assumptions and principles,

- the availability of a node is determined by the magnitude of its input flow only
- a node is shut down when the flow through it is either zero or below its predefined threshold value and its current capacity is non-zero; provided a shut down state is defined in its state-space diagram
- nodes are shut down in descending order of their rank
- nodes with zero input flow are placed higher on the scale.
- non-zero input flow nodes are ranked in order of their degree of inadequacy (i.e percentage by which the input falls short of the threshold)
- equally ranked nodes have the same priority and are shut down randomly.

Highlighted below is the iterative procedure proposed for system output calculation.

Step 1. Calculate system flow using equations 16, 19, 24 and 25.
Step 2. Find nodes requiring shut down and rank them.
Step 3. Select node at the top of the scale, save its next transition parameters, execute shut down and set its next transition time to infinity. If the input of the node is non zero, add it to the set **U**. Add the component to the maintenance list if transition to a maintenance state from shut down is possible and provided it's not already on the list and *force maintenance*.
Step 4. Repeat step 3 until all zero input nodes have been shut down. Go to step 6 if there's no node left to shut down.
Step 5. Repeat steps 1 to 3 until the list of components to shut down is exhausted.
Step 6. Determine state of sink nodes if necessary. This is the state number corresponding to the magnitude of flow into the node.

The complementary *restart* operation is carried out to restore nodes to their previous states prior to their being *shut down*. Enumerated below are the steps entailed.

Step 1. Perform system flow optimization using equations 16, 19, 24 and 25, previous sink indices and previous capacities of nodes before they were shut down.
Step 2. Select a node in shut down.
Step 3. Check if its input flow exceeds its threshold flow.
Step 4. If it does, determine how long it took in shut down.
Step 5. Restore node to its previous state and update its next failure time and state. Remove component from maintenance list if it's repairable only in shut down. Also remove component from **U** if applicable.
Next Failure Time = (Failure time before shut down) + (time spent in shut down)
Step 6. Repeat steps 2 to 5 until all nodes in shut down have been looped through. Go to step 8 if **U** is empty.
Step 7. Set to zero the current capacities of components in **U** and repeat steps 1 to 6.
Step 8. End procedure

3.3. Simulation Procedure

Summarised below are generic systematic steps proposed for simulation of structurally static homogeneous time dependent systems.

Step 1. Initialise system in preparation for simulation. This involves the following,
  a). initialization of vectors to save the current capacities; $\{c^{[i]}\}_{M\times 1}$, current sink indices; $\{\varepsilon^{[i]}\}_{M\times 1}$, next transition times;
  $\tau \mid \tau = \{\tau_i\}_{M\times 1}$ as well as state and output histories of nodes
  b). setting the required number of simulations ($Nsamples$), mission time (T_m)
Step 2. Set the simulation time; $t = 0$, $\mathbf{U} = \emptyset$, sample the next transition parameters of nodes and update τ.
Step 3. Determine initial state of sink nodes and carry out any necessary reconfigurations (including shut down and restart where necessary)
Step 4. Update initial state and output history of nodes
Step 5. Set the current simulation time to the minimum of τ. That is, $t = min(\tau)$.
Step 6. Check for nodes with next transition time equal to t i.e., $\tau_i = t$ and for each node, i,

a). effect the required transition and sample its next transition parameters
b). update τ, $\{c^{\{i\}}\}_{M\times1}$, $\{\varepsilon^{\{i\}}\}_{M\times1}$ and its state history
c). add component to maintenance list if new state is a maintenance state or if transition to a maintenance state from new state is possible.

Step 7. For each component on the maintenance list, *force maintenance*.

Step 8. Compare previous and current values of $\{c^{\{i\}}\}_{M\times1}$ and $\{\varepsilon^{\{i\}}\}_{M\times1}$. If a difference is observed in at least one of the two vectors,
a). restart any nodes in shut down
b). determine the new value of matrix $\mathbf{\Phi}$
c). calculate system flow and shut down nodes via the procedure described in section 3.2.2 using the new values of $\mathbf{\Phi}$, $\{c^{\{i\}}\}_{M\times1}$ and $\{\varepsilon^{\{i\}}\}_{M\times1}$. Note that all other parameters of equations 16, 19, 24 and 25 remain static throughout the simulation
d). for each node of the system, update output history if new output is different from output at previous transition.

Step 9. Repeat steps 5 to 8 until $t = T_m$, updating τ, $\{c^{\{i\}}\}_{M\times1}$, $\{\varepsilon^{\{i\}}\}_{M\times1}$ and node state & output histories at every transition.

Step 10. Repeat steps 2 to 9 for the desired number of times, saving node histories at each trial.

From the state and output histories of nodes, desired reliability and performance indices can be obtained using standard statistical analyses and the basic definition of the index of interest. These fall outside the scope of this paper but details can be found in [57].
Indices such as failure distribution, failure frequency, reliability function, average availability, instantaneous availability, instantaneous output, state probabilities, capacity factor and actual distributions of transitions with unknown distributions prior to the analysis are all obtainable from the state and output histories of nodes.

State duration based simulation technique achieves superior accuracy relative to the standard sequential Monte Carlo simulation described in [41] when applied to repairable systems. The upper hand is due to the incorporation of *shut down* and *restart* of nodes as a consequence of the behaviour of other system nodes. The standard technique on the other hand assumes statistical independence of system nodes.

3.4. Limitations

The proposed simulation procedure is challenged by two major limitations. The first is consequent of the assumptions used for *shut down* and *restart* of nodes. That is, the availability of a node is determined by the magnitude of its input flow only. This restricts the applicability of the methodology to homogeneous and independent heterogeneous systems. However, it can be easily extended to interdependent systems by incorporating fault trees and redefining the conditions for *shut down* and *restart*.

Also, the capacity constraint imposed on source and intermediate nodes means the effects of Common Cause Failures (CCF) resulting from flow redistribution cannot be studied. However, the procedure can be used in system design to estimate the required system parameter values to prevent these failures.

4. Case Studies

The principles and algorithms derived and described in the preceding sections were translated into a Matlab based application and applied to two case studies. The random variable generator available in the open source uncertainty quantification tool, *OpenCossan* [58]; developed at the Institute for Risk and Uncertainty of the University of Liverpool was incorporated to enhance sampling from any probability distribution including user-defined distributions.

4.1. Case Study 1: A Simple Pipe Network

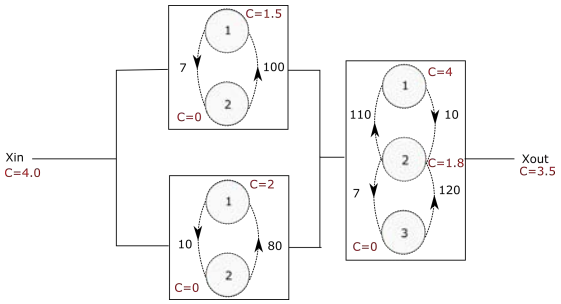


Figure 5: Structure of pipe network

Shown in Figure 5 is an oil transmission line showing the state-space representation of each component and was originally presented as Example 2.4 in [36]. Indicated beside each arc is the transition rate (in transitions per year) and beside each state number; the capacity (in tons of oil flow) of the component in that state. Flow is from left to right and the demand at Xout is conservatively assumed to be at 3.5 tons since this is the maximum possible flow through the network.

4.1.1. Analyses

The structure of the system can be represented by a network of 5 nodes. Numbering these chronologically from left to right, the network model is as given in Figure 6.

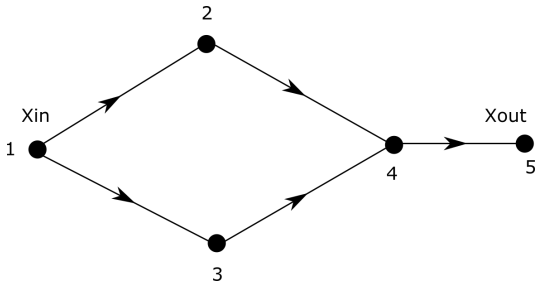


Figure 6: Network model of pipe network

Given that the efficiency and capacity of the links are of no interest in the analysis, the links are assumed to be 100% efficient and the adjacency matrix becomes;

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

From $\mathbf{A}$, $\mathbf{e}$ and $\mathbf{\Gamma}$ can be obtained using Algorithm 2 as;

$$\mathbf{e} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \\ 2 & 4 \\ 3 & 4 \\ 4 & 5 \end{pmatrix} \quad \mathbf{\Gamma} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{pmatrix}$$

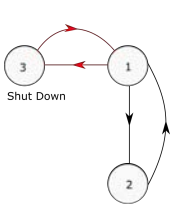
Applying the procedures described in Section 3.2.1, the parameters of the flow equations are obtained as;

$$\mathbf{\Theta} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathbf{ub} = \begin{pmatrix} 1.5 \\ 2 \\ 1.5 \\ 2 \\ 3.5 \end{pmatrix}$$

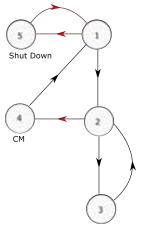
$$\mathbf{\Phi} = \begin{pmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{pmatrix}$$

The objective function, $\mathbf{\Psi}$ is;

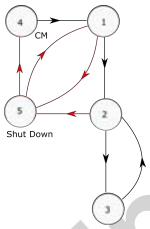
$$\mathbf{\Psi} = \begin{pmatrix} -1 & -1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} X_{12} \\ X_{13} \\ X_{24} \\ X_{34} \\ X_{45} \end{pmatrix}$$



(a) Nodes 2 & 3 in all 3 cases



(b) Node 4 in Case 2



(c) Node 4 in Case 3

Figure 7: State-space diagram of components

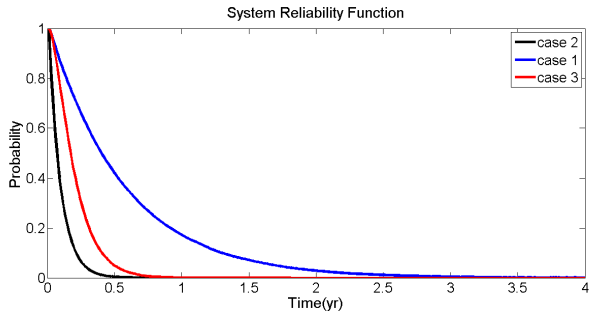
Note: *CM* represents corrective maintenance state. The problem under review was solved by Lisnianski et al.[36] using Markov Chain and assuming on-line repairs to node 4. This was repeated using the technique described in this paper and the analysis taken further by considering two other possible operating principles that can be imposed on the system. These are summarised as,

1. Assuming on-line repairs to node 4.
2. Node 4 is taken out of operation during repairs and these repairs commence almost instantaneously as the node enters a degraded state.
3. Node 4 is taken out of operation during repairs but a repair action only commences after the component is shut down as a result of failure of other nodes.

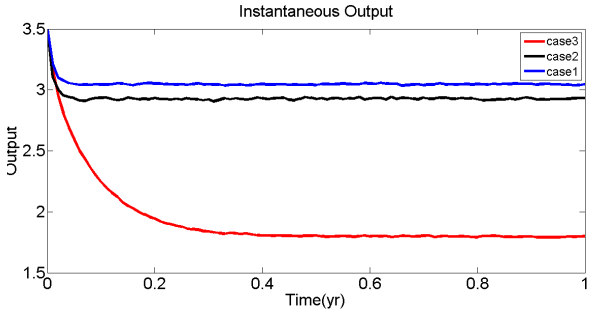
The system was analysed for these 3 cases and the results compared to bring out the effects of the various assumptions on the reliability and performance indices of the system. Figure 7 shows the state-space diagrams of all 3 components modelled as described in Section 2.1.

4.1.2. Results and Comments

For a mission time of 4 years and 20000 samples, the reliability and performance indices of the system obtained from the simulation are presented in Figure 8.



(a) Reliability function



(b) Instantaneous Output

Figure 8: System reliability and performance indices

Though this case study involves a simple system, it presents an irrefutable illustration of the effects of component modelling error on accurate estimation of system reliability and performance indices. As shown in Figure 8, modelling node 4 according to the assumptions made in case 1 would result in over estimation of system reliability and output if the system were operated according to case 2 or 3.

4.2. Case Study 2: A Multi-State Bridge Network

Bridge networks are a typical example of structural complexity exhibited by engineering systems. Reliability analysis of even the simplest bridge network is cumbersome when compared to analysis of a similarly sized system with a non-bridge configuration.

To show the applicability of the methodology developed, the imaginary multi-output, non-repairable complex bridge network shown in Figure 9 is considered. In the network, nodes 1, 2 and 3 respectively designated S_1, S_2 and S_3 are sources while nodes 4, 5 and 6 are sinks respectively labelled *Xout1*, *Xout2* and *Xout3*. The source nodes have identical failure characteristics but the capacity of S_1 is 1.5 times the individual capacities of the other sources. Also, the demand at node 6(70 units) is twice the individual demands at nodes 4 and 5. In addition, all diagonal links in the network are unidirectional with flow from left to right.

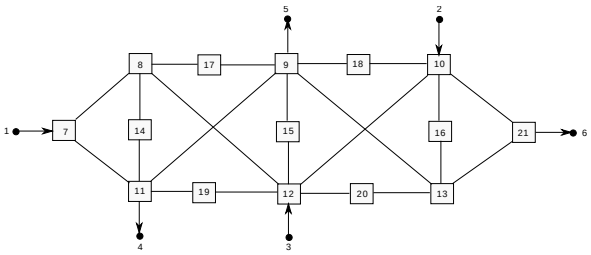


Figure 9: Block diagram of test bridge network

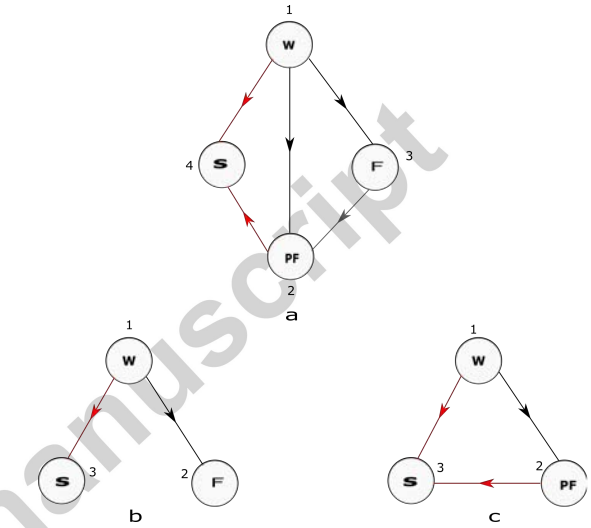


Figure 10: State space diagram of nodes (a)Source nodes (b)Intermediate nodes(c) Nodes 14-20 for case 3

4.2.1. Analyses

The intermediate nodes, according to their position in the system and similarity in failure characteristics are grouped and arbitrarily designated as follows,

a). nodes 7 to 13 and node 21 will be referred to as central nodes
b). nodes 14 to 16, referred to as vertical bridge nodes and
c). nodes 17 to 20, referred to as horizontal bridge nodes

Shown in Figure 10 are the state space diagrams of the nodes of the system as modelled by the procedure described in Section 2.1. *W*, *PF*, *F* and *S* respectively represent *Working*; depicting the operation of a node at its expected level, *partial failure*; when the node operates below its expected level, *Failure*; when the node is completely failed and *Shut down*; when the node is taken out of operation but not as result of failure. The network was analysed for three cases;

1. when flow through components 17 to 20 is from left to right only but components 14 to 16 are bidirectional
2. when components 14 to 20 are bidirectional
3. when components 14 to 20 are bidirectional and have a failure state in which their capacity remains unchanged but efficiency reduces by 20%.

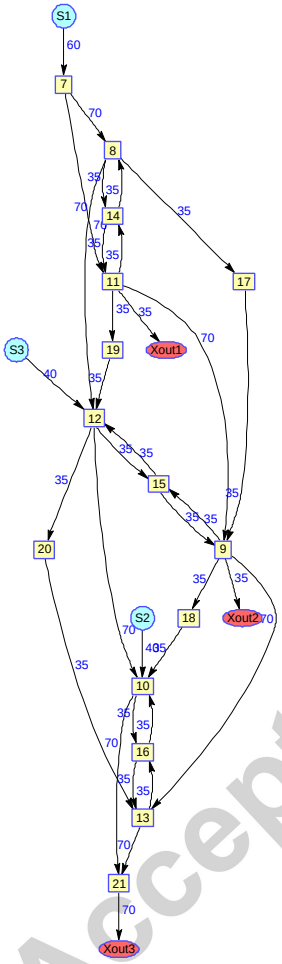


Figure 11: System network model for case 1

The properties and failure characteristics of the components of the system are presented in Table 1. $Exp(a)$ represents an exponential distribution of mean a, $LN(a,b)$; a Log-Normal distribution of mean a and standard deviation b and $W(a,b)$; a Weibull distribution with parameters a and b. It should be noted that the same component failure characteristics are used in all three cases.

Table 1: Node properties

Node Type	Transition	Distribution	Capacity
S_1	1-2 1-3 2-3	Exp(10) LN(20,2) LN(4.5,1.2)	$\begin{pmatrix} 60 & 45 & 0 & 0 \end{pmatrix}$
Central Nodes	1-2	LN(15,3)	$\begin{pmatrix} 70 & 0 & 0 \end{pmatrix}$
Bridge Nodes	1-2	W(12,2)	$\begin{pmatrix} 35 & 0 & 0 \end{pmatrix}$

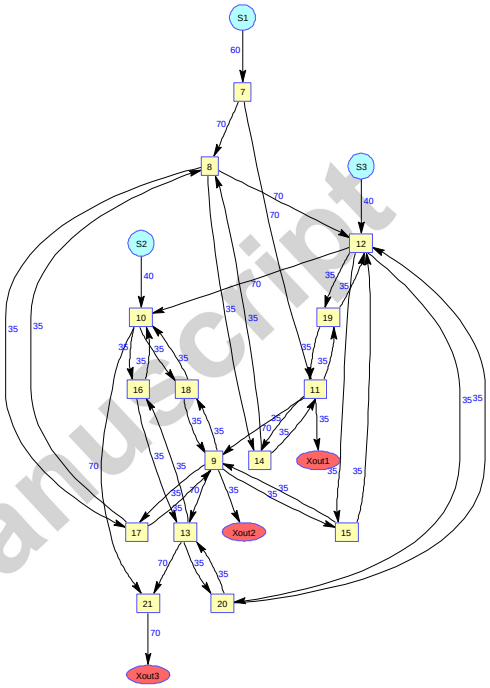
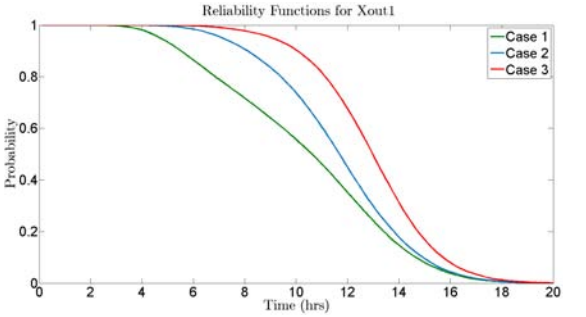


Figure 12: System network model for cases 2 & 3

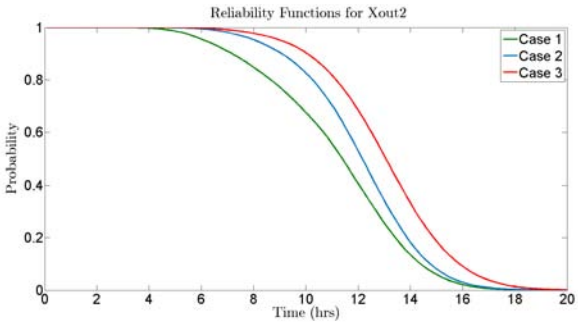
Each output node is modelled as a three-state object depending on the level of load flow realised at its input. These are, *Working*; assumed when the flow is at its maximum expected level, *partial failure*; when the flow is greater than zero but less than the required maximum and *failure*; assumed when no load flows into the node. Shown in Figures 11 and 12 are the network models of the system for all three scenarios under consideration with an indication of the maximum allowable load through each edge.

4.2.2. Results and Comments

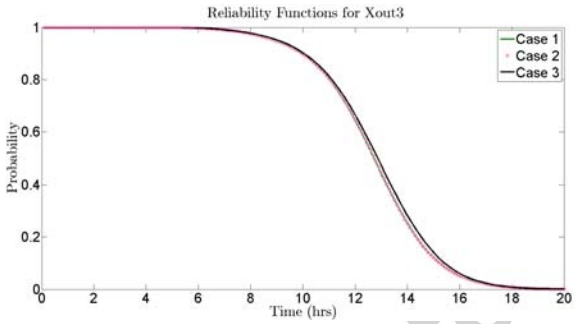
The system was simulated for a mission time of 20 hours using 60,000 samples. Presented in Table 2 is a summary of the average values of the most important reliability indices for $Xout1$ and shown in Figure 15 are its failure time distributions.



(a) *Xout*1

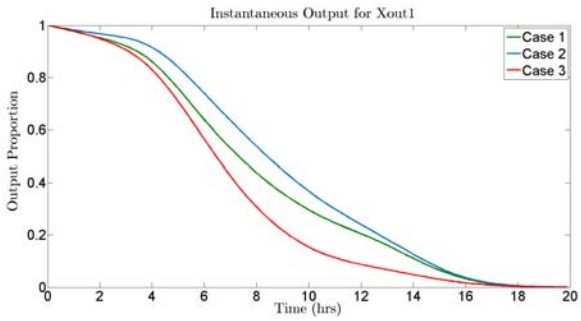


(b) *Xout*2

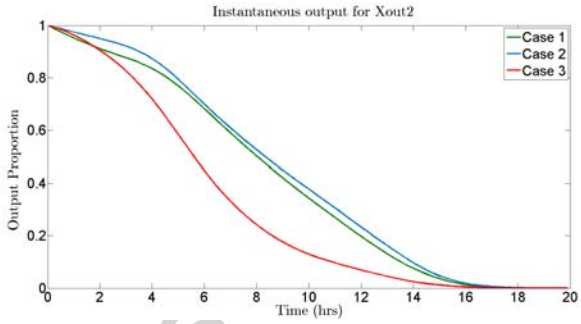


(c) *Xout*3

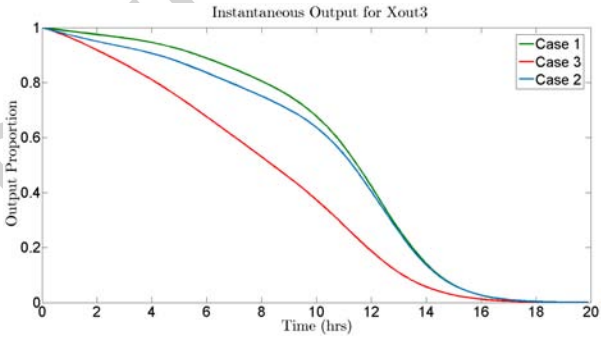
Figure 13: System reliability functions



(a) *Xout*1



(b) *Xout*2



(c) *Xout*3

Figure 14: Expected instantaneous system output

Table 2: Reliability indices for *Xout*1

Indices	**Case 1**	**Case 2**	**Case 3**
State Probability	0.1072 0.40398 0.4887	0.10737 0.46826 0.42437	0.10695 0.52755 0.3655
Capacity Factor	0.4345	0.44333	0.40592
Availability	0.5113	0.57563	0.6345
MTTF	10.2306	11.5172	12.6959
No. of Failures	0.9996	0.9996	0.9995

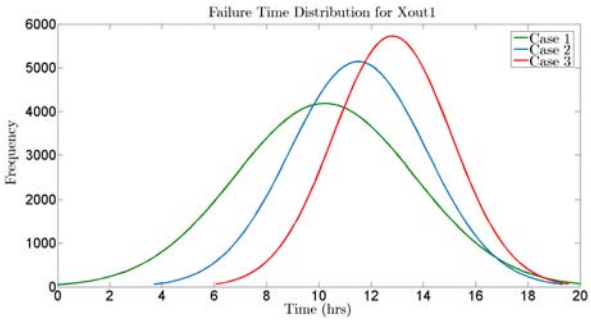


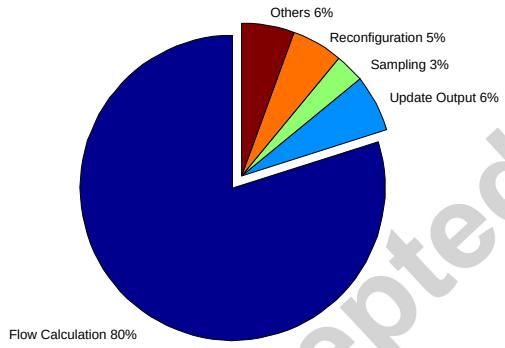
Figure 15: Failure time distribution for *Xout*1

The expected instantaneous output and reliability functions at the three output nodes are respectively presented in Figures 14 and 13. This case study was

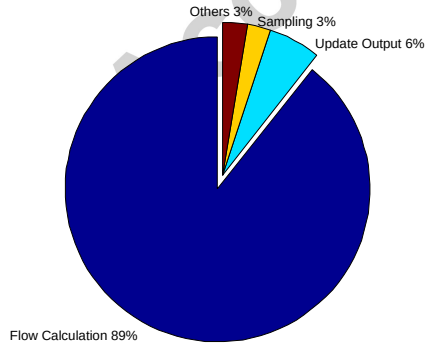
designed to portray the effects of system configuration and efficiency of components on the reliability and performance of output nodes. As evidenced in Figures 14 and 13, the performance and reliability of the system at output nodes except *Xout*3 are indeed affected by the factors under consideration.

System reliability in this case study is defined with respect to complete failure i.e., zero output. Therefore a non-zero load flow at a node renders it reliable irrespective of the magnitude of flow. The reliability of *Xout*3 remains unaffected in all three cases (see Figure 13c) because being a terminal node, the magnitude of flow realised at its inlet may change from case to case but the likelihood of the presence of this flow remains unaffected. The case study indicates why reliability should not be used as the only parameter to compare the response of a multi-state system under different conditions. It also shows that the response of an output node depends on its position in the system relative to other output nodes as well as sources.

4.3. Computational Costs



(a) Case Study 1



(b) Case Study 2

Figure 16: Simulation time utilisation per subroutine

Table 3: Actual computational cost per case study

	Case Studies	
	1	2
Average time per sample (s)	3.92	0.9287
Estimated total time (s)	78400	55722
Actual simulation time (s)	3163	2128.267
Improvement factor	24.79	26.18

Figure 16 shows how the total simulation time is distributed among the key tasks of the algorithm. They are based on a study of 1000 samples each, of the two case studies on a *48 core, 1895.257MHz AMD Opteron(tm) 6168 processor*. In Table 3 is a comparison of what the simulation times would be if run serially on a single core and the actual times spent using 24 cores running in parallel. We used 24 cores because the simulations were run on a shared facility and that was the maximum allowed for a single user. The estimated time is the product of the simulation time per sample and the total number of samples (20000 and 60000 for cases 1 and 2 respectively). Ideally, the improvement factor; the ratio of estimated to actual simulation time should be slightly less than the number of parallel cores due to initialization and overhead communication among cores. However, during parallelization, certain system initialization steps which were repeated for every sample in the serial simulation were carried out only once and broadcast across all 24 cores resulting in time gains.

Flow calculation, as depicted by Figure 16 accounts for the largest share of the computation time. Its magnitude depends on the nature and size of the system (i.e., whether the system is repairable or not), mission time and total number of simulation samples. There were on average 142 calls to the flow calculation subroutine per simulation sample in case study 1 with each call lasting 0.0221 seconds. Case study 2 had an average of 20 calls per simulation sample, each call lasting 0.0430 seconds. Incorporating variance reduction techniques resulting in less number of samples required should improve the time efficiency. Depending on the system, additional gains can also be derived from the complete omission of the *reconfiguration* subroutine. Reconfiguration (shut down & restart of nodes) is unnecessary for non-repairable systems (e.g. case study 2) and systems for which components are assumed to be statistically independent.

In summary, the size and degree of system activity determine the simulation time. However, as evidenced in Table 3, the 21-node non-repairable system with less activity required less simulation time than the 5-node

repairable system even though 3 times more simulation samples were used for the former. Therefore, the degree of system activity takes precedence over size in determining the simulation time.

5. Conclusions

Complex systems occur in various Engineering applications. However, their reliability analysis is normally impaired by their complexity and imposed operational loops. In the presence of these, a credible estimate of the associated reliability and performance indices is required. However, the assumptions applied to the system and component modelling process may negatively impact the integrity of the outcome obtained.

In this paper, a novel and generally applicable simulation approach to reliability and performance evaluation of complex multi-state systems with multiple outputs has been presented. The approach allows simulation of repairable and non-repairable complex multi-state systems of any topology without prior knowledge of system path and cut-sets. Owing to its flexibility, it can be easily extended to model cold and warm standby redundant systems, investigate cascading failure models and model systems with maintenance delays. The two numerical case studies presented have shown its effectiveness in analysing multi-state systems without the need to enumerate system performance levels or make unrealistic assumptions thereby rendering the process simple and robust to errors. The methodology enhances easy incorporation of system dynamics like flow losses, loading restrictions and promptness of maintenance actions in the reliability evaluation process as illustrated in the case studies.

Acknowledgements

The authors would like to acknowledge the gracious support of this work through the EPSRC and ESRC Centre for Doctoral Training on Quantification and Management of Risk & Uncertainty in Complex Systems & Environments.

References

[1] M. Yuchang, X. Liudong, S. V. Amari, J. B. Dugan, Efficient analysis of multi-state k-out-of-n systems, Reliability Engineering and System Safety.

[2] S. Distefano, A. Puliafito, Reliability and availability analysis of dependent-dynamic systems with drbds, Reliability Engineering and System Safety 94 (9) (2009) 1381–1393. URL http://dx.doi.org/10.1016/j.ress.2009.02.004

[3] G. Levitin, A. Lisnianski, Multi-state System Reliability Analysis and Optimization, in: Handbook of Reliability Engineering, Springer, 2003, Ch. 4, pp. 61–90.

[4] A. Lisnianski, G. Levitin, H. Ben-Haim, Structure optimization of multi-state system with time redundancy, Reliability Engineering and System Safety 67 (2) (2000) 103 – 112. URL http://dx.doi.org/10.1016/S0951-8320(99)00049-6

[5] G. Levitin, A. Lisnianski, H. Ben-Haim, D. Elmakis, Redundancy optimization for series-parallel multi-state systems, IEEE Transactions on Reliability 47 (2) (1998) 165 – 172. URL http://dx.doi.org/10.1109/24.722283

[6] I. Frenkel, A. Lisnianski, K. L, On the lz-transform application for availability assessment of an aging multi-state water cooling system for medical equipment, in: Applied Relaibility Engineering and Risk Analysis; Probabilistic Models and Statistical Inference, Wiley, 2014.

[7] W. Denson, History of reliability prediction, IEEE Transactions on Reliability 47 (3 -SP pt 2) (1998) 321 – 328.

[8] J. H. Saleh, K. Marais, Highlights from the early (and pre-) history of reliability engineering, Reliability Engineering & System Safety 91 (2) (2006) 249–256.

[9] M. Shooman, Reliability of computer systems and netwoks: fault tolerance, analysis and design, New York, NY, USA: Wiley, Inc, 2002.

[10] M. Abd-El-Barr, Design and analysis of reliable and fault-tolerant computer systems, Hackensack, NJ, USA: World Scientific Publishing Co., 2006.

[11] R. Billinton, N. Ronald, Reliability Evaluation of Engineering Systems, New York and London: Plenum Press, 1992, pp. 81–306.

[12] W. Wang, J. M. Loman, R. G. Arno, P. Vassiliou, E. R. Furlong, D. Ogden, Reliability block diagram simulation techniques applied to the ieee std. 493 standard network, Industry Applications, IEEE Transactions on 40 (3) (2004) 887–895, iD: 1.

[13] A. Volkanovski, M. Cepin, B. Mavko, Application of the fault tree analysis for assessment of power system reliability, Reliability Engineering and System Safety 94 (6) (2009) 1116 – 1127. URL http://dx.doi.org/10.1016/j.ress.2009.01.004

[14] W. Al-Khateeb, K. Al-Khateeb, S. Al-Irhayim, Availability evaluation of scalable complex networks, in: International Conference on Computer and Communication Engineering, ICCCE 2012, 2012, pp. 966–971. URL http://dx.doi.org/10.1109/ICCCE.2012.6271360

[15] S. Rai, K. Aggarwal, An efficient method for reliability evaluation of a general network, Reliability, IEEE Transactions on R-27 (3) (1978) 206–211. doi:10.1109/TR.1978.5220325.

[16] W.-C. Yeh, An improved sum-of-disjoint-products technique for the symbolic network reliability analysis with known minimal paths, Reliability Engineering & System Safety 92 (2) (2007) 260 – 268. doi:http://dx.doi.org/10.1016/j.ress.2005.12.006.

[17] S. Byun, I. Yang, M. G. Song, D. Lee, Reliability evaluation of steering system using dynamic fault tree, in: IEEE Intelligent Vehicles Symposium, 2013, pp. 1416 – 1420. URL http://dx.doi.org/10.1109/IVS.2013.6629665

[18] D. M. Shalev, J. Tiran, Condition-based fault tree analysis (cbfta): A new method for improved fault tree analysis (fta), reliability and safety calculations, Reliability Engineering and System Safety 92 (2007) 1231 – 1241. URL http://dx.doi.org/10.1016/j.ress.2006.05.015

[19] A. W. Al-Dabbagh, L. Lu, Reliability modeling of networked control systems using dynamic flowgraph methodology, Reliability Engineering and System Safety 95 (11) (2010) 1202 – 1209. URL http://dx.doi.org/10.1016/j.ress.2010.05.005

[20] H. Huang, J. Zhao, J. Tong, T. Liu, X. Guo, Reliability analysis

of digital ic system in nuclear power plant using dynamic flowgraph methodology, in: 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference, Vol. 3, Helsinki, Finland, 2012, pp. 1930 – 1938.

[21] M. Malhotra, K. S. Trivedi, Dependability modeling using petrinets, IEEE Transactions on Reliability 44 (3) (1995) 428 – 440. URL http://dx.doi.org/10.1109/24.406578

[22] M. Veeraraghavan, K. Trivedi, A combinatorial algorithm for performance and reliability analysis using multistate models, IEEE Transactions on Computers 43 (2) (1994) 229–234. doi:10.1109/12.262129.

[23] A. Lisnianski, Extended block diagram method for a multi-state system reliability assessment, Reliability Engineering & System Safety 92 (12) (2007) 1601 – 1607, special Issue on {ESREL} 2005. doi:http://dx.doi.org/10.1016/j.ress.2006.09.013.

[24] X. Zang, D. Wang, H. Sun, K. Trivedi, A bdd-based algorithm for analysis of multistate systems with multistate components, Computers, IEEE Transactions on 52 (12) (2003) 1608–1618. doi:10.1109/TC.2003.1252856.

[25] L. Xing, Y. Dai, A new decision-diagram-based method for efficient analysis on multistate systems, Dependable and Secure Computing, IEEE Transactions on 6 (3) (2009) 161–174. doi:10.1109/TDSC.2007.70244.

[26] W.-C. Yeh, A simple minimal path method for estimating the weighted multi-commodity multistate unreliable networks reliability, Reliability Engineering & System Safety 93 (1) (2008) 125 – 136. doi:http://dx.doi.org/10.1016/j.ress.2006.11.004.

[27] W.-C. Yeh, A fast algorithm for quickest path reliability evaluations in multi-state flow networks, Reliability, IEEE Transactions on 64 (4) (2015) 1175–1184. doi:10.1109/TR.2015.2452580.

[28] Y.-K. Lin, A simple algorithm for reliability evaluation of a stochastic-flow network with node failure, Computers & Operations Research 28 (13) (2001) 1277 – 1285. doi:http://dx.doi.org/10.1016/S0305-0548(00)00039-3.

[29] Y.-K. Lin, Using minimal cuts to evaluate the system reliability of a stochastic-flow network with failures at nodes and arcs, Reliability Engineering & System Safety 75 (1) (2002) 41 – 46. doi:http://dx.doi.org/10.1016/S0951-8320(01)00110-7.

[30] M. J. Zuo, Z. Tian, H.-Z. Huang, An efficient method for reliability evaluation of multistate networks given all minimal path vectors, IIE Transactions 39 (8) (2007) 811–817. arXiv:http://dx.doi.org/10.1080/07408170601013653, doi:10.1080/07408170601013653.
URL http://dx.doi.org/10.1080/07408170601013653

[31] W.-C. Yeh, An improved sum-of-disjoint-products technique for symbolic multi-state flow network reliability, Reliability, IEEE Transactions on 64 (4) (2015) 1185–1193. doi:10.1109/TR.2015.2452573.

[32] G. Levitin, Reliability evaluation for acyclic transmission networks of multi-state elements with delays, Reliability, IEEE Transactions on 52 (2) (2003) 231–237. doi:10.1109/TR.2003.813162.

[33] G. Yingkui, L. Jing, Multi-state system reliability: A new and systematic review, Procedia Engineering 29 (0) (2012) 531–536.

[34] G. Levitin, L. Xing, H. Ben-Haim, Y. Dai, Multi-state systems with selective propagated failures and imperfect individual and group protections, Reliability Engineering & System Safety 96 (12) (2011) 1657–1666.

[35] G. Levitin, The Universal Generating Function in Reliability Analysis and Optimization, Springer-Verlag London Limited, 2005.

[36] A. Lisnianski, I. Frenkel, Y. Ding, Multi-State System Reliability Analysis and Optimization for Engineers and Industrial Managers, Springer-Verlag London Limited, 2010.

[37] J.-J. Wang, C. Fu, K. Yang, X.-T. Zhang, G. hua Shi, J. Zhai, Reliability and availability analysis of redundant bchp (building cooling, heating and power) system, Energy 61 (2013) 531–540. URL http://dx.doi.org/10.1016/j.energy.2013.09.018

[38] A. Lisnianski, Y. Ding, Inverse lz-transform for a discrete-state continuous-time markov process and its application to multi-state system reliability, in: Applied Reliability Engineering and Risk Analysis, Wiley, 2014.

[39] A. Lisnianski, Lz-transform for a discrete-state continuous-time markov process and its applications to multi-state system reliability, in: Recent Advances in System Reliability; Signatures, Multi-state Systems and Statistical Inference, Springer, 2012.

[40] G. Levitin, A universal generating function approach for the analysis of multi-state systems with dependent elements, Reliability Engineering & System Safety 84 (3) (2004) 285 – 292. doi:http://dx.doi.org/10.1016/j.ress.2003.12.002.

[41] A. Sankarakrishnan, R. Billinton, Sequential monte carlo simulation for composite power system reliability analysis with time varying loads, IEEE Transactions on Power Systems 10 (3) (1995) 1540 – 1545.
URL http://dx.doi.org/10.1109/59.466491

[42] T. Solver, M. Amelin, State duration based monte carlo simulation model with independent failures for distribution system reliability analysis, in: 9th International Conference on Probabilistic Methods Applied to Power Systems, Stockholm, Sweden, 2006.
URL http://dx.doi.org/10.1109/PMAPS.2006.360213

[43] E. Zio, P. Baraldi, E. Patelli, Assessment of the availability of an offshore installation by monte carlo simulation, International Journal of Pressure Vessels and Piping 83 (4) (2006) 312 – 320. URL http://dx.doi.org/10.1016/j.ijpvp.2006.02.010

[44] M. Taheriyoun, S. Moradinejad, Reliability analysis of a wastewater treatment plant using fault tree analysis and monte carlo simulation, Environmental Monitoring and Assessment 187 (1).
URL http://dx.doi.org/10.1007/s10661-014-4186-7

[45] A. Ghaderi, M. Haghifam, S. M. Abedi, Application of monte carlo simulation in markov process for reliability analysis, in: IEEE 11th International Conference on Probabilistic Methods Applied to Power Systems, PMAPS 2010, Singapore, Singapore, 2010, pp. 293 – 298.
URL http://dx.doi.org/10.1109/PMAPS.2010.5528836

[46] S.-K. Au, J. L. Beck, Estimation of small failure probabilities in high dimensions by subset simulation, Probabilistic Engineering Mechanics 16 (4) (2001) 263 – 277.

[47] M. de Angelis, E. Patelli, M. Beer, Advanced line sampling for efficient robust reliability analysis, Structural Safety 52 (2015) 170 – 182.

[48] Z. Lu, S. Song, Z. Yue, J. Wang, Reliability sensitivity method by line sampling, Structural Safety 30 (6) (2008) 517 – 532.

[49] A. B. Huseby, B. Natvig, Discrete event simulation methods applied to advanced importance measures of repairable components in multistate network flow systems, Reliability Engineering & System Safety 119 (2013) 186 – 198. doi:http://dx.doi.org/10.1016/j.ress.2013.05.025.

[50] J.-A. Li, Y. Wu, K. K. Lai, K. Liu, Reliability estimation and prediction of multi-state components and coherent systems, Reliability Engineering & System Safety 88 (1) (2005) 93 – 98. doi:http://dx.doi.org/10.1016/j.ress.2004.07.010.

[51] W.-C. Yeh, Y.-C. Lin, Y. Chung, M. Chih, A particle swarm optimization approach based on monte carlo simulation for solving the complex network reliability problem, Reliability, IEEE Transactions on 59 (1) (2010) 212–221. doi:10.1109/TR.2009.2035796.

[52] A. Dwivedi, X. Yu, A maximum-flow-based complex network approach for power system vulnerability analysis, IEEE Transactions on Industrial Informatics 9 (1) (2013) 81 – 88.
URL http://dx.doi.org/10.1109/TII.2011.2173944

[53] M. T. Todinov, Analysis and optimization of repairable flow networks with complex topology, IEEE Transactions on Reliability 60 (1) (2011) 111 – 124.
URL http://dx.doi.org/10.1109/TR.2011.2104453

[54] Z. Chen, T. Berger, Reliability and availability analysis of manhattan street networks, IEEE Transactions on Communications 42 (-4) (1994) 511–522.

[55] S. Mehrotra, On the implementation of a primal-dual interior point method, SIAM Journal on Optimization 2 (4) (1992) 575–601. doi:10.1137/0802028.
URL http://dx.doi.org/10.1137/0802028

[56] M. Kojima, S. Mizuno, A. Yoshise, A primal-dual interior point algorithm for linear programming, in: N. Megiddo (Ed.), Progress in Mathematical Programming, Springer New York, 1989, pp. 29–47.

[57] A. Lisnianski, I. Frenkel, Y. Ding, Multi-State System Reliability Analysis and Optimization for Engineers and Industrial Managers, Springer-Verlag London Limited, 2010, Ch. Multi-State System Reliability and Its Measures, pp. 16–27.

[58] E. Patelli, M. Broggi, M. D. Angelis, M. Beer, Opencossan: An efficient open tool for dealing with epistemic and aleatory uncertainties, in: Vulnerability, Uncertainty, and Risk: Quantification, Mitigation, and Management - Proceedings of the 2nd International Conference on Vulnerability and Risk Analysis and Management, ICVRAM 2014 and the 6th International Symposium on Uncertainty Modeling and Analysis, ISUMA 2014, 2014, pp. 2564 – 2573.
URL http://dx.doi.org/10.1061/9780784413609.258